LLNL-TR-408156

# An Adaptive Derivative-based Method for Function Approximation

C. Tong

October 24, 2008

## Disclaimer

This document was prepared as an account of work sponsored by an agency of the United States government. Neither the United States government nor Lawrence Livermore National Security, LLC, nor any of their employees makes any warranty, expressed or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States government or Lawrence Livermore National Security, LLC. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States government or Lawrence Livermore National Security, LLC, and shall not be used for advertising or product endorsement purposes.

This work performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract DE-AC52-07NA27344.

LLNL-TR-XXXX
Unlimited Release
October 2008

# An Adaptive Derivative-based Method for Function Approximation [*]

Charles Tong
Lawrence Livermore National Laboratory
Livermore, CA 94550-0808

### Abstract

To alleviate the high computational cost of large-scale multi-physics simulations to study the relationships between the model parameters and the outputs of interest, response surfaces are often used in place of the exact functional relationships. This report explores a method for response surface construction using adaptive sampling guided by derivative information at each selected sample point. This method is especially suitable for applications that can readily provide added information such as gradients and Hessian with respect to the input parameters under study. When higher order terms (third and above) in the Taylor series are negligible, the approximation error for this method can be controlled. We present details of the adaptive algorithm and numerical results on a few test problems.

## 1 Introduction

A typical simulation model takes in a set of input (or parameter) values and returns some outputs. For deterministic models we can define the input-output relationships in the form of $F : X \rightarrow Y$. Here, $X \in \Re^m$ represents the m-dimensional parameter space and $Y$ represents the n-dimensional space containing all possible output values. A simulation model must undergo rigorous verification, validation and uncertainty quantification before they can be used with confidence. These rigorous procedures generally require a large number of simulation runs. For models that are themselves computationally expensive (many hours on hundreds or more processors), such studies can become prohibitive. One solution to this

[*] This work was performed under the auspices of the U.S. Department of Energy Lawrence Livermore National Laboratory under Contract No. DE-AC52-07NA27344.

problem is the use of computer experiments to construct functions, called a *response surfaces*, that closely approximate the original functions and which are inexpensive to evaluate.

In this report we describe an adaptive method to efficiently construct an approximate function for a given simulation model. This method is especially suitable for applications that can readily provide added information such as gradients and the Hessian matrix. Without such additional information, one can still provide finite difference-based gradients and Hessian. When third and higher order terms in the Taylor series are negligible, the approximation errors of the response surface can be assessed readily.

There are two major steps in constructing an approximation for a particular function. The first step is *experimental design*, or choosing a limited number of points in the parameter space at which to evaluate the original function. The selected sample points and corresponding output constitute a *training sample*. During this step, it is important that the training data capture the maximum amount of information with respect to the input-output relationships of the original function. Without any prior knowledge about the function (for example, monotonicity, maximum/minimum), the parameter space should be uniformly sampled. The second step is to apply function approximation schemes to create an approximate function based on the training sample.

Much work has been performed in function approximation. Some of the most popular methods are linear and nonlinear regressions, tensor products of piecewise polynomials, multi-variate adaptive splines (MARS) [3], artificial neural network (ANN) [2], support vector machine (SVM) [1], and Gaussian process (GP or kriging) [5]. In this report we propose another method based on using derivative information to guide adaptive sampling and interpolation. Our proposed method assumes the existence of the first and second derivatives at every point in the parameter space, and constructs m-dimensional regions of validity (or ROVs) to cover the whole parameter space. The sizes of the ROVs are governed by the computed curvatures (second derivatives) and the prescribed tolerance. The actual evaluation of any point not in this training set is replaced by locating the ROV it belongs to and the using the ROV data (output value and its derivatives) for interpolation. The challenge in this approach is how to efficiently and effectively determine the training set. We investigate the use of adaptive sampling guided by the derivative information.

In Section 2, we describe in detail an algorithm to construct our derivative-based response surfaces. We present some preliminary numerical results in Section 3. We conclude with a few remarks in Section 4.

# 2 An Algorithm for Response Surface Construction

Consider again a function $Y = F(X)$ where $X \in \Re^m$ and $Y \in \Re^n$. We also assume that the function $F$ is twice differentiable everywhere in the domain $\Omega$. Suppose that the function has been evaluated at the point $X = X^*$. Then another point in the proximity of $X^*$ can be expressed by the following Taylor expansion:

$$F(X^* + \Delta X) = F(X^*) + \Delta X \cdot G(X)|_{X=X^*} + 0.5\Delta X^T H(X)|_{X=X^*}\Delta X + \text{ higher order terms}$$

where $G(X)$ and $H(X)$ are the gradient vector and the Hessian matrix, respectively. The idea behind our scheme is to approximate the function in the neighborhood of $X^*$ using the first two terms of the Taylor expansion. Assuming that the third and higher order terms are negligible, then the error of approximation can be estimated by the second order terms. Given an error tolerance acceptable to users, the size of the region (ROV) where errors are sufficiently accurate can be determined. The algorithmic framework for this construction is as follow:

```
Algorithm RS1:
Choose initial point $X_0$. Define tolerance $\epsilon$
Evaluate $F(X_0), G(X_0), H(X_0)$ and define its ROV.
M = 0
while (the parameter space is not fully covered) do
    M = M + 1
    Choose the next point $X_M$
    Evaluate $F(X_M), G(X_M), H(X_M)$ and define new ROV
end
```

A few details of the algorithms need to be worked out:

1. How to define each ROV?

2. How to determine complete coverage of the parameter space?

3. How to choose the initial evaluation point?

4. How to select the next evaluation point?

5. How to ensure continuity across ROVs?

Each ROV is defined by the coordinate of its data point $X$; the corresponding $F(X)$, $G(X)$; and the radius of the hypersphere given by

$$r = \sqrt{\frac{2\epsilon}{\lambda_{\max}}}$$

where $\lambda_{\max}$ is the largest eigenvalue (in magnitude) of $H(X)$.

To decide whether the whole parameter space is fully covered, we use a large space-filling test sample (for example, quasi-Monte Carlo sample; or factorial sample, if the input dimension is small). During the course of the algorithm, we check to make sure every point in the test sample resides in one of the ROVs.

The initial evaluation point can be the center of the $\Omega$, or any of the test sample points. After the construction of a ROV, each test sample point should be examined to see if it belongs to the new ROV and its estimation errors should be updated. The test sample point with the largest error is selected to be the next evaluation point.

There are two other issues with the current function approximation scheme: (1) how to interpolate points sitting inside multiple ROVs; and (2) how to ensure continuity of interpolation across ROVs. First, the algorithm does not preclude constructing ROVs that are overlapping. Assigning points inside the overlapped region to one of the ROVs may result in discontinuity in the response surface. Even if there is no overlap between adjacent ROVs, discontinuity will still exist. We have devised a scheme to solve the continuity problem based on weighted sum. Specifically, if a data point resides in $p$ ROVs, then the interpolated value is given by

$$Y = \frac{\sum_{i=1}^{p} w_i Y_i}{\sum_{i=1}^{p} w_i}.$$

The weights are defined as

$$w_i = \begin{cases} \frac{\epsilon - 0.5 r_i^2 \lambda_{\max}}{0.5 r_i^2 \lambda_{\max}} & r_i < R_i \\ 0 & \text{otherwise} \end{cases}$$

where $r_i$ is the distance between the data point and the $i$-th ROV's evaluation point $X_i$ and $R_i$ is the radius of the hypersphere covered by the $i$-th ROV. If the data point coincides with the evaluation points in one of the ROVs (say ROV $k$), then its value is set to $Y_k$.

Incorporating the details gives rise to the following modified algorithm:

```
Algorithm RS2:
Create a test sample $S$ of size $N$ (a large but reasonable number)
Form an array $C(1:N)$ and set $C(1:N) = 0$ (all sample points are not covered)
Form an array $E(1:N)$ and set $E(1:N) = \infty$ (infinite initial error)
Define tolerance $\epsilon$. $i = 0$.
while ($S$ is not fully covered or not all $C(j) = 1; j = 1:N$) do
        i = i + 1
        Select evaluation point $X_i$ (from the test sample $S$).
        Evaluate $F(X_i), G(X_i), H(X_i)$ and construct ROV $i$.
        Update interpolation errors $E(1:N)$ for all points in $S$.
        Update $C(1:N)$ (if point $j$ resides in this ROV, set $C(j) = 1$.)
        If $C(j) = 1$ for all $j = 1, .., N$, exit.
        Choose the next point $X_{i+1}$ in $S$ based on maximum error $E(1:N)$.
end
```

The number of model evaluations needed in our method is $n_r(\frac{m(m+3)}{2} + 1)$ where $n_r$ is the number of ROVs created. Hence, our method is suitable when the number of input dimension is not too large. However, additional knowledge about the model can help to reduce the computational cost. For example, for additive functions, the Hessian is a diagonal matrix and hence the number of model evaluations needed becomes $n_r(2m+1)$. For functions that are tensor products of polynomial functions, the off-diagonal elements of the Hessian can be constructed from the diagonal elements and hence only $n_r(2m+1)$ model evaluations are needed.

# 3 Numerical Experiment

We first investigated the performance of our algorithm on a simple test problem:

$$Y = \sum_{k=1}^{m} (x_k - 0.5)^2, \quad x_k \in [0,1],$$

which has a diagonal Hessian matrix with each diagonal element equal to 2. We compute the gradients and Hessian using finite differences.

When $m = 2$ and given a tolerance $\epsilon = 0.05$, we can calculate that each region has a area of $\pi\epsilon$. In the bounded box $[0,1]^2$, the theoretical minimum number of regions to cover the box (assuming circular regions) is $\frac{1}{\pi\epsilon} \approx 7$. Similarly, for $m = 3$ and $m = 4$, the minimum number of regions are 22 and 82, respectively. In practice, many of these regions can overlap and so we do not expect we can achieve these minimum numbers. These numbers are for reference only. In the following we focus on the comparison between with and without refinement. Since the number of parameters is small, we use the factorial designs in all cases.

We use the METIS ([9]) sample of size 50000. The initial point $X_0$ is the first point in the sample. The gradients and Hessian matrix are evaluated using finite differences with a width of $10^{-5}$.

The number of ROVs to achieve the tolerance for the three cases are $19, 94$ and 407, respectively. These numbers are factors of $3-5$ from the optimal, but the optimal is hard to achieve because it is difficult to prevent overlaps between the hyperspheres.

Next, we investigated a more difficult example ([7]):

$$Y = (0.8 * R + 0.35 * \sin(\frac{2.4 * \pi * R}{\sqrt{(2.0)}})) * (1.5 * \sin(1.3 * \theta)) \tag{1}$$

where $R = \sqrt{X_1^2 + X_2^2}$, $\theta = \tan^{-1} \frac{X_2}{X_1}$ and $X_1, X_2 \in [0,1]$. Figure 1 describes the input-output relationship (created by using the actual function).

We again used the METIS design with sample size 50000 and we experimented with two different $\epsilon$'s, 0.1 and 0.01. The number of regions needed is 37 and 299 with a total of 222 and 1794 model evaluations, respectively.

In Figure 2 we plot the locations of the sample points for the two thresholds. We observe from the $\epsilon = 0.01$ plot that most of the points are concentrated near the area where the gradients are relatively large, demonstrating the adaptivity of this algorithm.

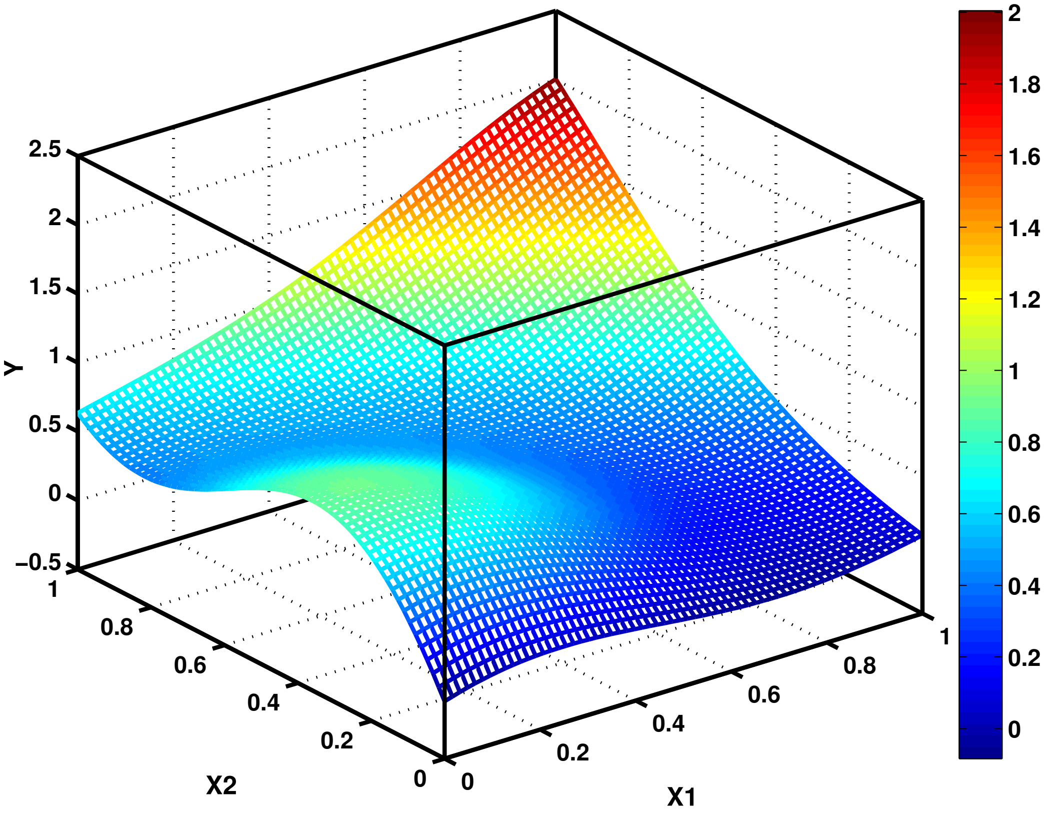

Figure 1: Plot of Equation 1

The response surfaces for the two thresholds are plotted in Figure 3. We also compare our adaptive method with a MARS-based method ([3] using LP-$\tau$ sampling method with the same corresponding sample sizes. The MARS-based response surfaces are given in Figure 4. We observe that while tightening the threshold for our derivative-based method improves the quality of the response surface, increasing the sample size seems to make little or no improvement for this test problem.

# 4 Final Remarks and Conclusion

In this report we describe an adaptive method for efficiently constructing approximate functions for simulation models. This method is especially suitable for applications where derivative information is readiy available. We have combined the derivative-based idea with multiresolution sampling techniques to design a response surface algorithm. Preliminary results show that it is a viable approach. Future effort will focus on further improving the efficiency of this method and comparing it against other response surface approach in efficiency and accuracy.

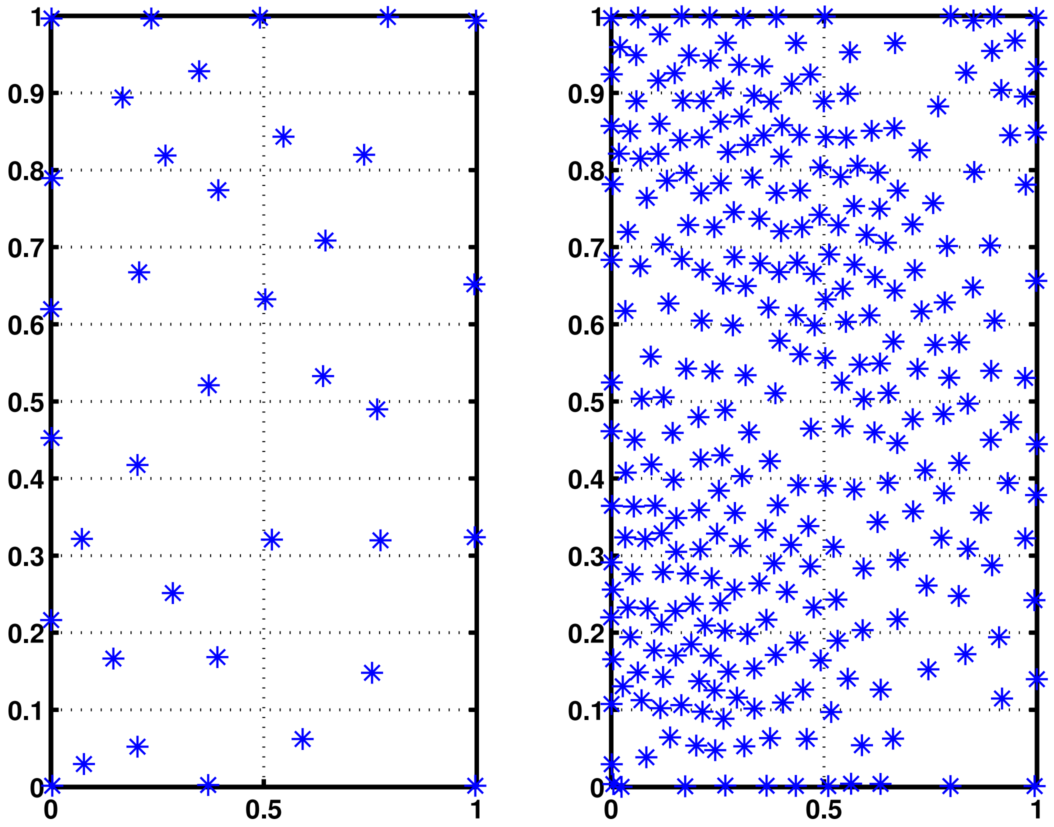

Figure 2: Scatter plots of Equation 1: (left) $\epsilon = 0.1$, (right) $\epsilon = 0.01$

**Acknowledgment:**

The author would like to thank Radu Serban for initiating this work and implementing the first prototype based on Monte Carlo sampling.

# References

[1] S. M. Clarke, J. H. Griebsch, and T. W. Simpson, *Analysis of Support Vector Regression for Approximation of Complex Engineering Analyses,* Journal of Mechanical Design, Nov 2005, Vol. 127, pp. 1077-1087.

[2] T. Draelos and D. Hush, *A constructive neural network algorithm for function approximation, IEEE International Conference on Neural Networks*, 1, 50-55, 1996.

[3] J. H. Friedman, *Multivariate adaptive regression splines, Annals of Statistics* 19.1, 1-141, 1991.

[4] A. S. Hedayat, N. J. a. Sloane, and John Stufken, *Orthogonal Arrays: Theory and Applications,* Springer Series in Statistics, 1999.

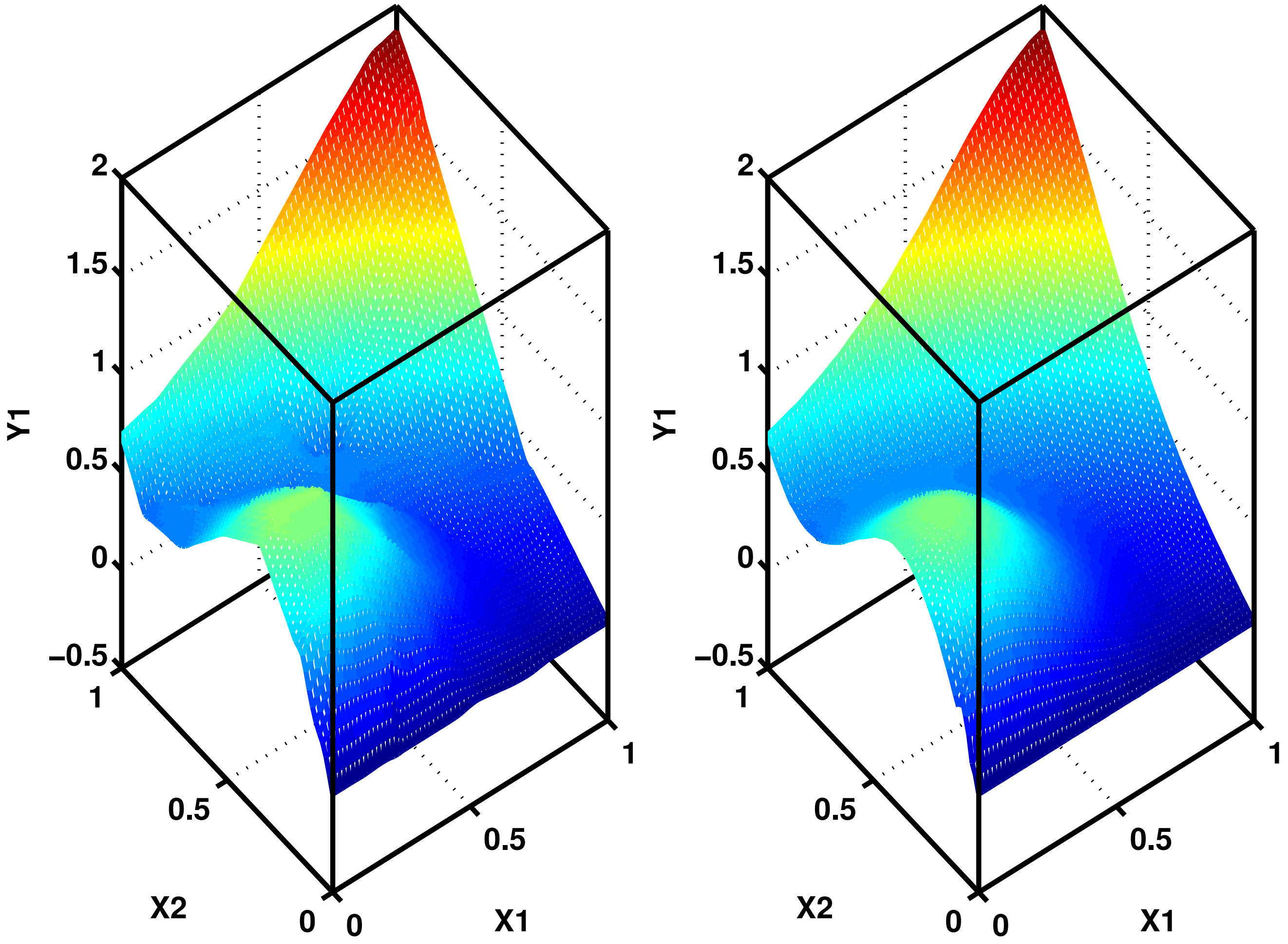

Figure 3: Response surfaces using the new method: (left) $\epsilon = 0.1$, (right) $\epsilon = 0.01$

[5] D. J. C. MacKay, *Gaussian Processes - a Replacement for Supervised Neural Networks,* Lecture Notes at NIPS, 1997.

[6] R. H. Myers and D. C. Montgomery, *Response Surface Methodology,* Second Edition, Wiley Series in Probability and Statistics, 2002.

[7] V. J. Romero, T. Krishnamurthy and L. P. Swiler, *Progressive Response Surfacecs,* Proceeding of the 9th ASCE Specialty Conference on Probabilistic Mechanics and Structural Reliability.

[8] C. Tong, *Refinement Strategies for Stratified Sampling Methods,* Reliability Engineering and System Safety 91 (2006), Issue 10-11, pp 1257-1265.

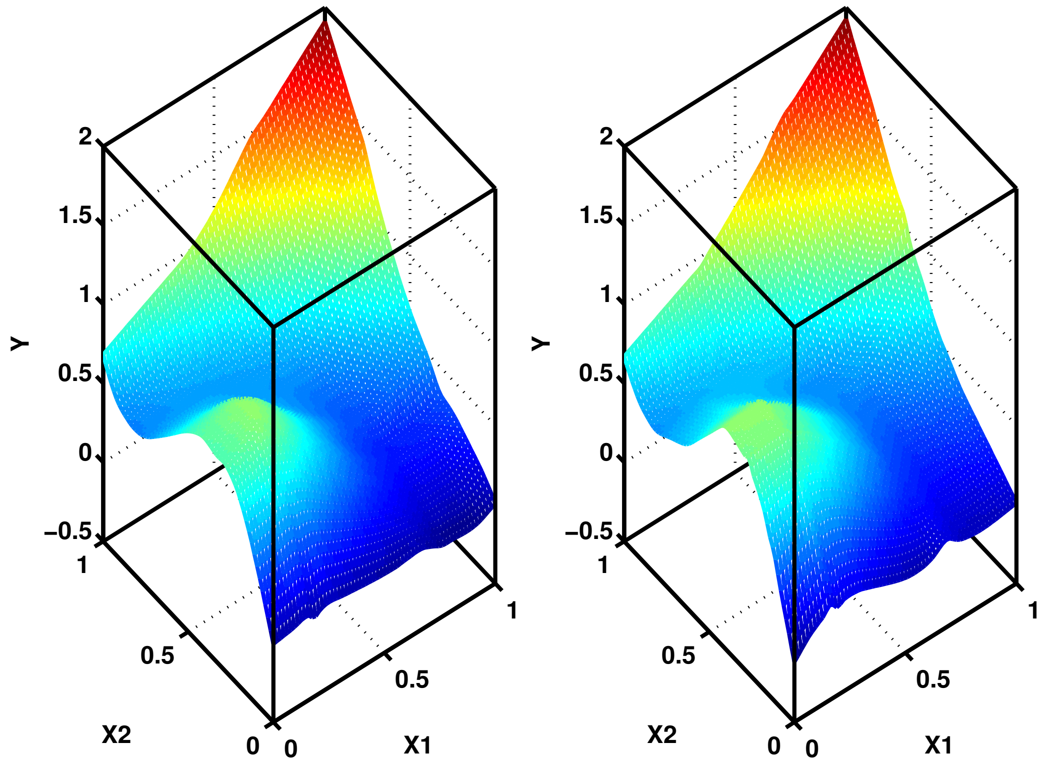

Figure 4: Plot of Equation 1 using MARS

[9] G. Karypis and V. Kumar, *A fast and high quality multilevel scheme for partitioning irregular graphs,* International Conference on Parallel Processing, pp. 113-122, 1995.